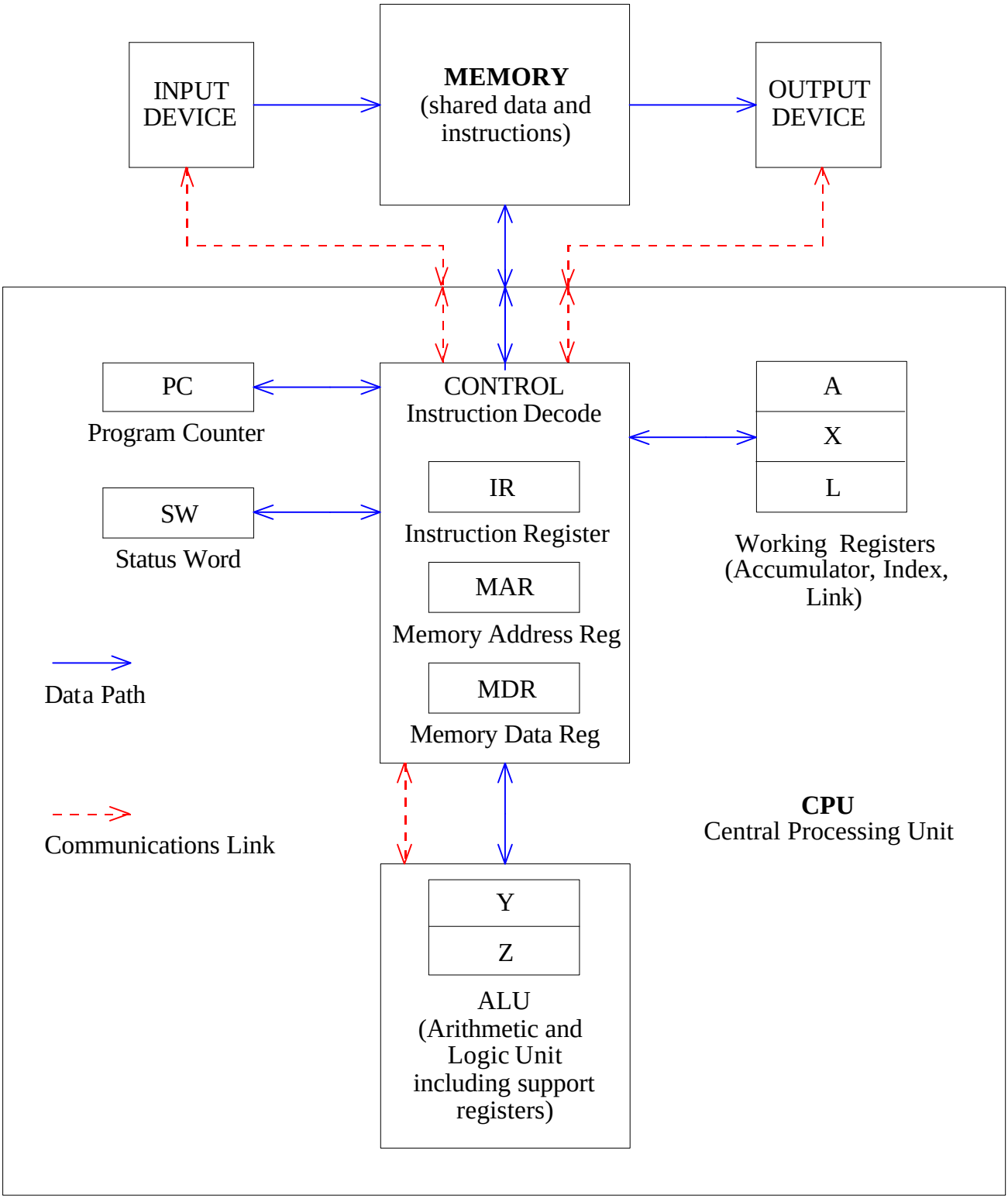
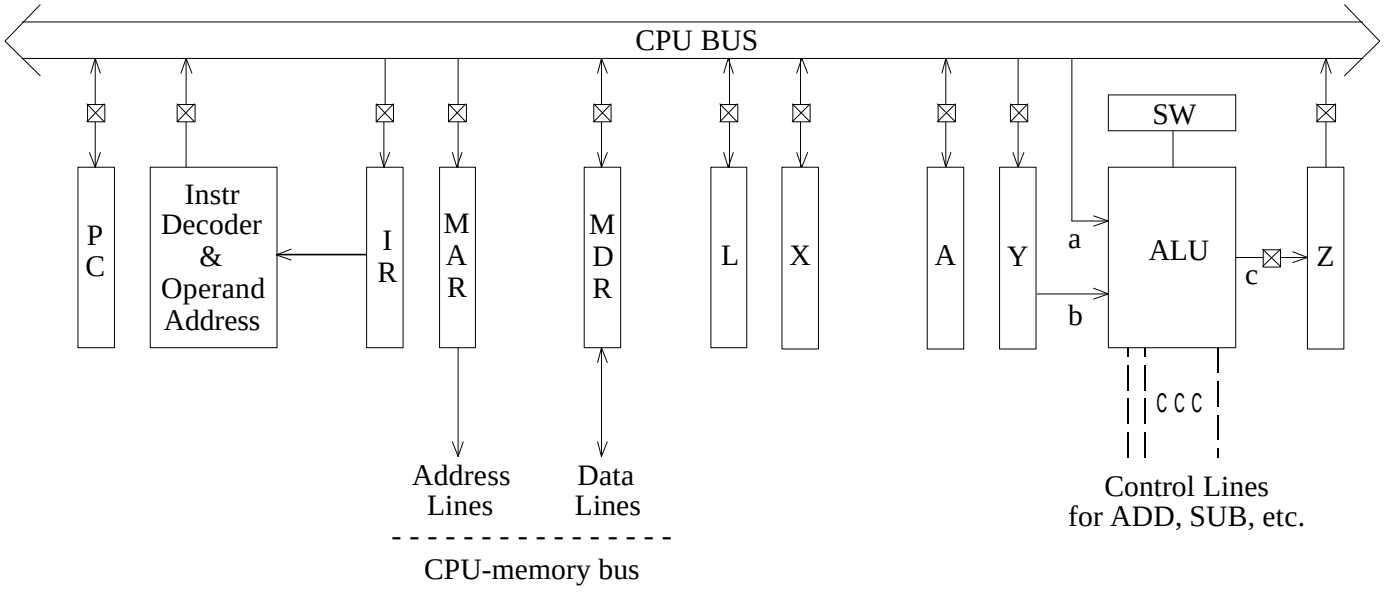
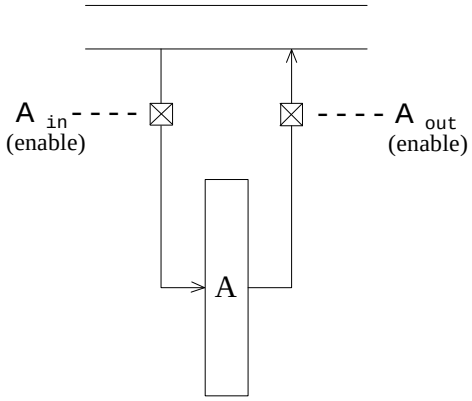
Simple SIC Hardware Organization

(separate I/O:memory bus and CPU:memory bus)



Single Bus CPU Architecture for Implementing Simple SIC





Register-Bus Gating

Data transfer example: Y←A

A_{out}, Y_{in}

Gating Signals:

IC_{out}, IC_{in}, $Addr_{out}$, IR_{in}, MAR_{in}, MDR_{out}, MDR_{in},
A_{out}, A_{in}, X_{out}, X_{in}, L_{out}, L_{in}, Y_{in}, Z_{out}, Z_{in}

Memory I/O control signals:

`Read`, `Write`, `WaitM` (Wait for Memory)

ALU commands:

`Add`, `Sub`, `Set carry` (to 1), `ShiftR Y`, `ShiftL Y`, `Clear Y`,
`Compare`, `GT`, `LT`, `EQ`, `NE`

A full microcode sequence for the ADD instruction to accumulate in A:

Instruction fetch

IC_{out}, MAR_{in}, Read, Clear Y, Set carry, Add, Z_{in}

Z_{out}, IC_{in}, WaitM

MDR_{out}, IR_{in}

Operand fetch

$Addr_{out}$, MAR_{in}, Read

A_{out}, Y_{in}, WaitM

Accumulate

MDR_{out}, Add, Z_{in}

Z_{out}, A_{in}

End

Each line of microcode represents those signals which are "on". All others are presumed to be "off".

The ALU command "Compare" activates a comparison of the value in register Y against the value from the bus and sets the lead bit of the status word SW to 0 for the case "=" and 1 for the case "≠". If the case is "≠", the next bit is also set, with 1 0 indicating "<" Y and 1 1 indicating ">" Y.

This can be used to construct a machine language "compare to A" instruction

COMP <addr>

to compare the contents of register A against the data at the referenced address, setting the status word.

The ALU instructions EQ, LT, GT, and NE operate as follows:

For any of SW=1 0 and LT, SW=1 1 and GT, SW=0 - and EQ, SW=1 - and NE, ALU input from line "b" is routed to ALU output "c" and otherwise ALU input from line "a" is routed to ALU output "c"; i.e., LT, GT, EQ, and NE cause either the bus (input a) or register Y (input b) to be routed to the ALU output (output c) depending on the current value in the status word (SW). LT, GT, EQ, and NE allow the construction of conditional branch instructions.

The routing patterns are summarized by the following diagram:

	SW (after Compare)	ALU routing on EQ	NE	LT	GT
	0 0	Y → Z	bus → Z	bus → Z	bus → Z
=	0 1	Y → Z	bus → Z	bus → Z	bus → Z
≠	1 0 (bus < Y)	bus → Z	Y → Z	Y → Z	bus → Z
≠	1 1 (bus > Y)	bus → Z	Y → Z	bus → Z	Y → Z

For example, assuming the status word was set by an earlier instruction (such as COMP), a microcode sequence such as "jump on greater than"

JGT <addr>

which causes a branch (or jump) to the referenced address if SW=1 1 can be constructed. Its format might be as follows:

Instruction fetch	IC_{out}, MAR_{in}, Read, Clear Y, Set carry, Add, Z_{in}
	Z_{out}, IC_{in}, WaitM
	MDR_{out}, IR_{in}
Set branch options	$Addr_{out}$, Y_{in}
	IC_{out}, GT, Z_{in}
Set branch	Z_{out}, IC_{in}
	End